AMENDMENTS TO THE SPECIFICATION

Please replace the paragraph beginning at page 2, line 17, with the following rewritten paragraph:

-- The disadvantages of layered systems leading to performance inefficiencies consist primarily of overhead, both in computation and in message headers, caused by the abstraction barriers between layers. Because a message often ~~have~~ has to pass through as many as 10 or more protocol layers on its way from a host to the network and from the network to a host, the overhead produced by the boundaries between the layers is often more than the actual computation being done. Different systems have reported overheads for crossing layers of up to 50μs. Therefore, it is highly desirable to mitigate the disadvantages and to develop techniques that reduce delays by improving performance of layered protocols. --

Please replace the paragraph beginning at page 2, line 30, with the following rewritten paragraph:

--Several methods have been suggested to improve performance of layered communication protocols. One of the methods is described

by Robbert van Renesse in the article "Masking the Overhead of Protocol Layering", Proc. of the Proceedings of the 1996 ACM SIGCOMM Conference, September 1996, which article is incorporated herein by reference. In that article protocols are optimized through the use of a protocol accelerator which employs, among others, such optimization techniques as pre- and post- processing of a message in order to move computation overhead out of the common path of execution. The use of that method led to the successful reduction of communication latency, but not the computation. Pre- and post-processing was done through a layering model where handlers were broken into the operations to be done during and after messaging operations (preprocessing for the next message is appended to the post-processing of the current message). The protocol accelerator also used small connection identifiers in order to compress headers from messages and message packing techniques in order to achieve higher throughput. The use of the protocol accelerator achieved code latencies of 50μs for protocol stacks of 5 layers. The total time required for pre- and post- processing of one message during send and receive operations is approximately 170μs, with a header overhead of 16 bytes. This result is an improvement in comparison to code-latencies of 26μs in Ensemble, protocol

headers of 8 bytes, and total processing overhead for a *receive* operation followed by a *send* operation of 63μs, with a protocol stack that has more than twice as many layers. --

Please replace the paragraph beginning at page 5, line 8, with the following rewritten paragraph:

--It is ~~another~~ an object of an embodiment of the present invention to achieve optimization of performance of layered protocols by selecting a "basic unit of optimization." To achieve optimization, the method automatically extracts a small number of common sequences of operations occurring in protocol stacks. These common sequences are called "event traces". ~~The~~ That embodiment of the invention provides a facility for substituting optimized versions of these traces at runtime to improve performance. These traces are amenable to a variety of optimizations that dramatically improve performance. The traces can be mechanically extracted from protocol stacks. Event traces are viewed as orthogonal to protocol layers. Protocol layers are the unit of development in a communication system, they implement functionality related to a single protocol. Event traces, on the other hand, are the unit of execution. Therefore, ~~the~~ an

embodiment of the present invention focuses on event traces to optimize execution. --

Please replace the paragraph beginning at page 5, line 27, with the following rewritten paragraph:

--It is ~~yet another~~ an object of an embodiment of the present invention to provide optimized protocols of high performance which are easy to use. Normally, the protocol optimizations are made after-the-fact to already working protocols. This means that protocols are designed largely without optimization issues in mind. In this embodiment of the present invention, optimizations require almost no additional programming, only a minimal amount of annotation of the protocol layers is necessary (the annotation consists of marking the start and end of the common paths of the source code). Therefore, optimizations can call for annotating only small portions of the protocols which belong to the common path, reducing the complexity of the optimization techniques. In addition, the optimizations of this embodiment of the present invention place few limitations on the execution model of the protocol layers. --

Please replace the paragraph beginning at page 6, line 9, with the following rewritten paragraph:

--It is ~~also~~ an object of an embodiment of the present invention to be able to apply the current state of verification technology to small, layered protocols which are just within the range of current verification technologies, whereas large, monolithic protocols are certainly outside this range. --

Please replace the paragraph beginning at page 16, line 1, with the following rewritten paragraph:

--The second ~~pass~~ step is used to eliminate intermediate data structures. The second step removes the explicit use of events in the protocol layers. In the described layering model, event records are used to pass information between protocol layers in a stack. These records contain temporary information about a message, which information follows the message through the layers and event queues. Each event must be allocated, initialized, and later released. It is not necessary to use events explicitly, because event traces encompass the life of the initial event and all spawned events. Therefore, the contents of the event record

can instead be kept in local variables within the trace handler. Compilers are often able to place such variables in registers. --

Please replace the paragraph beginning at page 16, line 32, with the following rewritten paragraph:

--The fourth step is to apply traditional optimizations to the trace handlers. This operation proves to be very effective, because the previous ~~passes~~ steps create large basic blocks which compilers can optimize. Furthermore eliminating constant folding and dead-code also proved to be effective due to the elimination of event records. For instance, if one protocol layer marks an event record's field with some flag to cause an operation to happen at another layer, the ~~flap~~ flag can be propagated through the trace handler so that the ~~flap~~ flag is never set at the first layer or checked at the second layer. --

Please replace the paragraph beginning at page 17, line 13, with the following rewritten paragraph:

-- The second class of optimizations provided by the present invention reduces the size of message headers. The protocol

layers in a stack prepend their headers to a message as it moves up or down the protocol stack. Later the message headers are stripped off ~~by popped off~~ by the peer layers at the destination host. To facilitate optimization these headers are divided into three classes, two of which are suitable for compression. --

Please replace the paragraph beginning at page 23, line 25, with the following rewritten paragraph:

-- It is important to note that since the time the test results represented in Fig. 5 were obtained, ~~significance~~ significant improvements in ML compilers made it possible to achieve the performance of the optimized pure ML protocol stack similar to that of the C protocol. --